(19)日本国特許庁（ＪＰ）　(12) **公開特許公報（Ａ）**　(11)特許出願公開番号

特開平10－191090

(43)公開日　平成10年(1998)７月21日

(51)Int.Cl.[6]	識別記号	ＦＩ		
Ｈ０４Ｎ　1/60		Ｈ０４Ｎ	1/40	Ｄ
Ｂ４１Ｊ　2/525		Ｇ０９Ｇ	5/06	
Ｇ０６Ｔ　1/00		Ｂ４１Ｊ	3/00	Ｂ
Ｇ０９Ｇ　5/06		Ｇ０６Ｆ	15/66	Ｎ
Ｈ０４Ｎ　1/46				３１０

審査請求　未請求　請求項の数10　ＯＬ　（全 14 頁）　最終頁に続く

(21)出願番号　特願平9－274697

(22)出願日　平成９年(1997)10月７日

(31)優先権主張番号　特願平8－282726

(32)優先日　平８(1996)10月24日

(33)優先権主張国　日本（ＪＰ）

(71)出願人　000002369
セイコーエプソン株式会社
東京都新宿区西新宿２丁目４番１号

(72)発明者　深沢　賢二
長野県諏訪市大和３丁目３番５号　セイコーエプソン株式会社内

(72)発明者　笠原　広和
長野県諏訪市大和３丁目３番５号　セイコーエプソン株式会社内

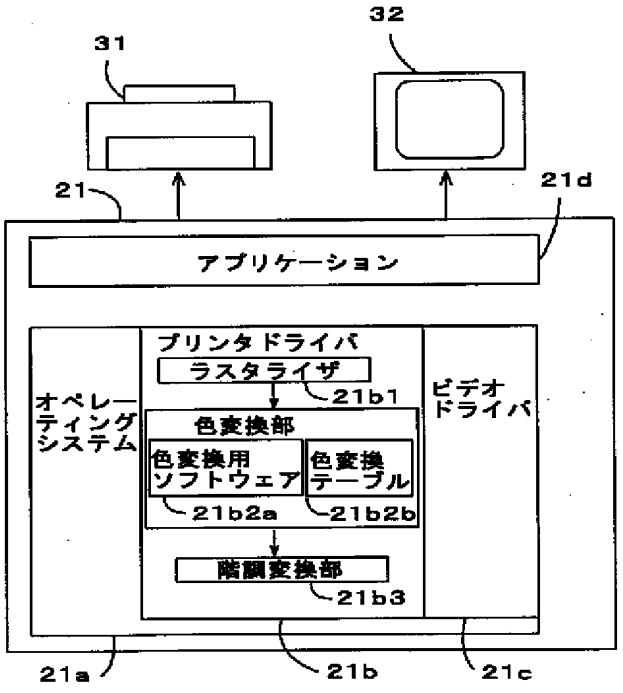
(74)代理人　弁理士　鈴木　喜三郎　（外２名）

(54)【発明の名称】　色変換テーブルの製造装置及び製造方法並びに記録媒体

(57)【要約】　（修正有）

【課題】　プリンタドライバで使用する色変換テーブルはユーザの環境によっては最適なものとは限らなかった。

【解決手段】　画像処理装置を構成するコンピュータ２１にてインストーラが実行されると、元色変換テーブルから色変換テーブル２１ｂ２ｂを生成するが、このときの格子点増加処理では、ラグランジュの補間公式を利用した非線形補間演算で格子点を増加させたり、線形補間で格子点を増加させるなどし、また、その際に固定した格子点の数であっても良いし環境や入力画像に応じた格子点の数としてもよく、小さなサイズの元色変換テーブルから適切なサイズの色変換テーブル２１ｂ２ｂを生成することができる。



1

【特許請求の範囲】

【請求項１】　異なる表色空間の間で階調表色データを変換するために変換元の表色空間に複数の格子点を設定し、この格子点に変換先の表色空間での階調表色データを対応させた色変換テーブルを生成する色変換テーブルの製造装置であって、

少数の格子点において変換の対応関係を記憶する元色変換テーブルと、

この元色変換テーブルの格子点を補間演算によって増加させて変換に利用する色変換テーブルを生成する補間手段とを具備することを特徴とする色変換テーブルの製造装置。

【請求項２】　上記請求項１に記載の色変換テーブルの製造装置において、上記補間手段は、複数の格子点の対応関係から非線形補間演算で補間する非線形補間演算手段を具備することを特徴とする色変換テーブルの製造装置。

【請求項３】　上記請求項２に記載の色変換テーブルの製造装置において、上記元色変換テーブルは、均等な間隔の格子点となっていることを特徴とする色変換テーブルの製造装置。

【請求項４】　上記請求項１に記載の色変換テーブルの製造装置において、上記補間手段は、複数の格子点の対応関係から線形補間演算で補間する線形補間演算手段を具備することを特徴とする色変換テーブルの製造装置。

【請求項５】　上記請求項１～請求項４に記載の色変換テーブルの製造装置において、上記補間手段は、補間で増加させる格子点の数を選択可能としていることを特徴とする色変換テーブルの製造装置。

【請求項６】　上記請求項５に記載の色変換テーブルの製造装置において、上記補間手段は、増加させる格子点の数を環境に応じて設定することを特徴とする色変換テーブルの製造装置。

【請求項７】　上記請求項５に記載の色変換テーブルの製造装置において、上記補間手段は、増加させる格子点の数を変換画像の種類に応じて設定することを特徴とする色変換テーブルの製造装置。

【請求項８】　上記請求項１に記載の色変換テーブルの製造装置において、上記色変換テーブルはコンピュータにて参照するとともに、通常時は同コンピュータの補助記憶装置に上記元色変換テーブルを記憶するとともに、同色変換テーブルの参照実行時に上記コンピュータの主記憶領域に展開することを特徴とする色変換テーブルの製造装置。

【請求項９】　異なる表色空間の間で階調表色データを変換するために変換元の表色空間に複数の格子点を設定し、この格子点に変換先の表色空間での階調表色データを対応させた色変換テーブルを生成する色変換テーブルの製造方法であって、

少数の格子点において変換の対応関係を記憶する元色変換テーブルの前記格子点を補間演算によって増加させて変換に利用する色変換テーブルを生成することを特徴とする色変換テーブルの製造方法。

2

【請求項１０】　異なる表色空間の間で階調表色データを変換するために変換元の表色空間に複数の格子点を設定し、この格子点に変換先の表色空間での階調表色データを対応させた色変換テーブルをコンピュータにて生成する色変換テーブル作成プログラムを記録した記録媒体であって、

少数の格子点において変換の対応関係を記憶する元色変換テーブルの前記格子点を補間演算によって増加させて変換に利用する色変換テーブルを生成することを特徴とする記録媒体。

【発明の詳細な説明】

【０００１】

【発明の属する技術分野】本発明は、異なる表色空間の間で階調表色データを変換するために変換元の表色空間での格子点に変換先の表色空間での階調表色データを対応させた色変換テーブルを生成する色変換テーブルの製造装置および製造方法並びに色変換装置に関する。

【０００２】

【従来の技術】従来、この種の色変換テーブルとして、コンピュータ上のカラー画像をカラー印刷するカラー印刷システムが知られている。

【０００３】コンピュータの内部では、カラー画像は縦横に並べられた各画素ごとについて赤緑青の三原色（Ｒ，Ｇ，Ｂ）で階調表示されているが、一般のカラー印刷装置においてはシアン、マゼンダ、イエローの三色（Ｃ，Ｍ，Ｙ）あるいはこれにブラックを加えた（Ｃ，Ｍ，Ｙ，Ｋ）四色で階調表示のない状態で印刷される。従って、カラー印刷するためには赤緑青の三原色（Ｒ，Ｇ，Ｂ）の表示からシアン、マゼンダ、イエローの三色（Ｃ，Ｍ，Ｙ）の表示への色変換の作業と、階調表示から階調のない表示への階調変換の作業が必要となる。なお、色空間自体は一つの空間であるものの、座標の取り方によって表示が異ならざるをえないため、以下においては、便宜上、座標の取り方に応じた表色空間と呼ぶことにする。

【０００４】この（Ｒ，Ｇ，Ｂ）表示から（Ｃ，Ｍ，Ｙ）表示への色変換は変換式によって一義的に定まるものではなく、それぞれの階調を座標とする色空間について相互に対応関係を求めておき、この対応関係から逐次変換するのが通常である。ここにおいて、少なくとも変換元の（Ｒ，Ｇ，Ｂ）表示が各色について２５６階調であったとすれば、約１６７０万個（２５６×２５６×２５６）の要素の色変換テーブルを持たなければならない。

【０００５】効率的な記憶資源の利用を考えた結果、すべての座標値についての対応関係を用意しておくのではなく、適当なとびとびの格子点について対応関係を用意

しておき、補間演算を併用するようにしている。すなわち、（Ｒ，Ｇ，Ｂ）表色空間の中でのある座標の色について（Ｃ，Ｍ，Ｙ）表色空間の対応関係を求めるときには同座標を取り囲む格子点の対応関係を利用し、線形補間などの演算を経て同座標の対応関係を求めている。

【０００６】このような色変換テーブルは、一般にプリンタドライバが備えており、色変換テーブルを含めたプリンタドライバ自体は個々のカラー印刷装置に対応して一つだけが提供されている。従って、色変換テーブルについても、記憶資源との対比から適当に定められた格子点の数に特定されていた。

【０００７】

【発明が解決しようとする課題】上述した従来の色変換テーブルにおいては、プリンタドライバを提供する側が一般的な記憶資源との対比に基づいて色変換テーブルを作成しているため、必ずしもユーザの環境によっては最適なものとは限らないという課題があった。すなわち、ユーザの環境によってはまだまだ大きいという場合もあるし、より大きなサイズとした方が良いという場合もあった。

【０００８】さらに、色変換テーブルの大きさによって印刷品質にも差が生じるため、一定の大きさの色変換テーブルでは十分ではないという課題もあった。

【０００９】本発明は、上記課題にかんがみてなされたもので、ユーザの環境などに最適な色変換テーブルを生成することが可能な色変換テーブルの製造装置および製造方法の提供を目的とする。

【００１０】

【課題を解決するための手段】上記目的を達成するため、請求項１にかかる発明は、異なる表色空間の間で階調表色データを変換するために変換元の表色空間での格子点に変換先の表色空間での階調表色データを対応させた色変換テーブルを生成する色変換テーブルの製造装置であって、少数の格子点において変換の対応関係を記憶する元色変換テーブルと、この元色変換テーブルの格子点を補間演算によって増加させて変換に利用する色変換テーブルを生成する補間手段とを具備する構成としてある。

【００１１】上記のように構成した請求項１にかかる発明においては、もともと存在しているのが格子点を少数とした元色変換テーブルであり、記憶資源をわずかしか必要としていない。そして、補間手段はこの元色変換テーブルの格子点を補間演算によって増加させて色変換テーブルを生成し、この生成された色変換テーブルを色変換に利用する。

【００１２】元色変換テーブルにおける格子点の数は、生成される色変換テーブルと比較して相対的に少なければよく、必ずしも極めて少ない数値である必要はない。特に、三次元の格子点であれば、格子点の多少は３乗で変化してくる。すなわち、格子点を半分にするだけで色変換テーブルのサイズは１／８の大きさとなり、少しの差でも記憶資源に対して大きな効果を与えることになる。

【００１３】補間演算手段については各種の演算を適用することが可能であり、その一例として、請求項２にかかる発明は、請求項１に記載の色変換テーブルの製造装置において、上記補間手段は、複数の格子点の対応関係から非線形補間演算で補間する非線形補間演算手段を具備する構成としてある。

【００１４】上記のように構成した請求項２にかかる発明においては、補間手段の非線形補間演算手段が複数の格子点の対応関係から非線形補間演算で格子点を補間する。

【００１５】非線形補間演算を行うことにより、増加される格子点での対応関係が正確に再現されることになる。従って、元色変換テーブルの格子点を少なくしても非常に再現性の良い結果を得ることが可能となる。

【００１６】一方、元色変換テーブルの格子点の間隔についてはこの補間演算にも関連があり、その一例として、請求項３にかかる発明は、請求項２に記載の色変換テーブルの製造装置において、上記元色変換テーブルは、均等な間隔の格子点となった構成としてある。

【００１７】非線形補間演算においては必ずしも格子点間隔が均等である必要はないが、不均等な格子間隔であるとすると演算式の係数が複雑になってくる。そして、少なくとも三次元の立方体内で補間演算で格子点を増加させようとする場合には、各軸方向毎に複数の格子点から中間的な格子点を求めるような作業が必要になり、このような場合に演算式の係数が複雑になってくると補間演算が煩雑になり、作業が煩わしい。これに対して均等な格子点間隔である場合には一部の係数などが一定となってループ処理も適用しやすくなるなどの状況が生じる。

【００１８】むろん、非線形補間演算以外の他の補間演算によっては均等である方が好ましい場合もあるし、不均等である方が好ましい場合もあり、補間演算に応じて適宜変更可能である。

【００１９】また、非線形補間演算以外の演算を利用する一例として、請求項４にかかる発明は、請求項１に記載の色変換テーブルの製造装置において、上記補間手段は、複数の格子点の対応関係から線形補間演算で補間する線形補間演算手段を具備する構成としてある。

【００２０】線形補間の場合には演算式自体が複雑でないというメリットもあるし、演算に必要な格子点が軸方向に対して二点であるという性質がある。従って、対応関係が大きく変化する部分においては格子点を細かくすることにより、正確な対応関係を容易に得られるようになるし、逆に、対応関係がさほど変化しない部分においては格子点を粗くするといったことも容易になる。

【００２１】補間演算手段が増加させる格子点の数は、

必ずしも一定である必要はなく、その一例として、請求項５にかかる発明は、請求項１～請求項４に記載の色変換テーブルの製造装置において、上記補間手段は、補間で増加させる格子点の数を選択可能とした構成としてある。

【００２２】増加される格子点の数により色変換テーブルのサイズが変化するし、補間演算によっては格子点が色変換での変換精度に影響を与える場合もある。従って、補間で増加させる格子点の数が選択可能であることにより、ユーザの環境に対して最適な格子点の数とすることが可能となってくる。

【００２３】この場合において、請求項６にかかる発明は、請求項５に記載の色変換テーブルの製造装置において、上記補間手段は、増加させる格子点の数を環境に応じて設定する構成としてある。

【００２４】格子点の数によって色変換テーブルのサイズが変化するし、色変換時のヒット率も変わってくる。さらには、増加させる格子点の位置によっては補間演算を容易にさせたりすることにもなる。従って、環境に応じて総合的に格子点を増加させることにより、最適な色変換を提供できる。環境に応じて増加する格子点を選択する方針として、記憶資源が多ければ大きなサイズの色変換テーブルとすればよいし、格子点が多いほど色変換精度が高い場合において色変換精度を高くすることを望むのであれば格子点を多くするようにすればよいし、補間演算時に２の累乗での乗除算が容易であるようならば、格子点の間隔が２の累乗となるようにすればよい。

【００２５】さらに、同様に増加する格子点を選択するにあたり、請求項７にかかる発明は、請求項５に記載の色変換テーブルの製造装置において、上記補間手段は、増加させる格子点の数を変換画像の種類に応じて設定する構成としてある。

【００２６】上述したように、格子点が多いほど色変換精度が高い場合があり、そのような場合において画像によっては色変換精度を高く要求するものもあればさほど必要としないようなものもある。このため、変換画像の情報が色変換精度を高く要求するようなものであれば格子点をできるだけ多くし、色変換精度をさほど要求しないようなものであれば格子点をさほど多くしない。この場合の方針として、例えば、オペレーティングシステムなどから変換画像の種類を知ることができるような場合に、ファイルの拡張子がビットマップであれば写真などの色情報の重要度が高いものと判断して格子点をできる限り多くする一方、ファイルの拡張子がドローデータやビジネスグラフを指すような場合には色情報の重要度は低いものと判断して格子点をさほど増やさないといったことが有効である。

【００２７】発明の思想の具現化例における他の一例として、請求項９にかかる発明は、異なる表色空間の間で階調表色データを変換するために変換元の表色空間に複数の格子点を設定し、この格子点に変換先の表色空間での階調表色データを対応させた色変換テーブルを生成する色変換テーブルの製造方法であって、少数の格子点において変換の対応関係を記憶する元色変換テーブルの前記格子点を補間演算によって増加させて変換に利用する色変換テーブルを生成することを特徴としておる。

【００２８】すなわち、必ずしも実体のある装置に限らず、その方法としても有効であることに相違はない。

【００２９】ところで、このような色変換テーブルを備える色変換装置は単独で存在する場合もあるし、ある機器に組み込まれた状態で利用されることもあるなど、発明の思想としてはこれに限らず、各種の態様を含むものである。従って、ソフトウェアであったりハードウェアであったりするなど、適宜、変更可能である。

【００３０】その一例として、印刷インクに対応した表色空間に対して異なる表色空間の階調表色データを変換するにあたり、少数の格子点において変換の対応関係を記憶する元色変換テーブルを使用し、格子点を補間演算によって増加させて色変換テーブルを生成し、この生成した色変換テーブルを利用して色変換する構成とすることもできる。

【００３１】すなわち、プリンタドライバは印刷インクに対応した表色空間に対して異なる表色空間の階調表色データを変換するために、色変換テーブルを参照することになるが、この際に少数の格子点の元色変換テーブルから補間して格子点を増し、格子点を増加した色変換テーブルを使用して色変換する。

【００３２】発明の思想の具現化例として色変換装置のソフトウェアとなる場合には、請求項１０のように、かかるソフトウェアを記録した記録媒体上においても当然に存在し、利用されるといわざるをえない。むろん、その記録媒体は、磁気記録媒体であってもよいし光磁気記録媒体であってもよいし、今後開発されるいかなる記録媒体においても全く同様に考えることができる。また、一次複製品、二次複製品などの複製段階については全く問う余地無く同等である。その他、ソフトウェアである場合にインストール作業において上述したような格子点を増加させる処理を行うことも可能であるし、供給方法として通信回線を利用して行なう場合でも本発明が利用されていることにはかわりないし、提供する側はソフトウェア提供装置として機能するものであり、同様に本発明を利用していることに相違ない。

【００３３】さらに、一部がソフトウェアであって、一部がハードウェアで実現されている場合においても発明の思想において全く異なるものはなく、一部を記録媒体上に記憶しておいて必要に応じて適宜読み込まれるような形態のものとしてあってもよい。さらには、かかる色変換テーブル使用することになるカラーファクシミリ機やカラーコピー機においても適用可能であることはいうまでもない。

【００３４】
【発明の効果】以上説明したように本発明は、小さなサイズの元色変換テーブルから格子点を増した色変換テーブルを生成するものであるため、色変換をしない状態では最低限の記憶資源しか必要としない一方、色変換を行うときには必要な大きさとなった色変換テーブルとすることができ、より融通性の高い色変換テーブルの製造装置を提供することができる。むろん、必要なときにだけ展開して不要なときには展開しないようにしても良いし、記憶資源に余裕があるのであれば展開した状態のまま残しておくこともできる。
【００３５】また、請求項２にかかる発明によれば、非線形補間演算で格子点を増加させるようにしているため、増加された格子点の精度が高くなり、良好な色変換結果を得ることができる。これは、逆にいえばより格子点を少なくしても良好な結果を得ることができるという効果にもなる。
【００３６】さらに、請求項３にかかる発明によれば、元色変換テーブルの格子点を均等な間隔とすることにより非線形補間演算が複雑化しないようにでき、演算時間などの減少を図ることができる。
【００３７】さらに、請求項４にかかる発明によれば、演算量の少ない線形補間を利用して簡易に補間演算を行うことができる。また、線形補間の簡易さを利用すれば大きな変化部分で格子点を密にすることにより、簡易でありながら精度の向上を図ることができるなどのメリットもある。
【００３８】さらに、請求項５にかかる発明によれば、増加させる格子点の数を選択できるので、ユーザの記憶資源に合わせて色変換テーブルのサイズを決めるなど、より柔軟にユーザの環境に適した色変換テーブルを生成することができるようになる。これはまた、複数のサイズの色変換テーブルを生成することもできるので、必要に応じて適宜選択して利用するといったことも可能となる。
【００３９】さらに、請求項６にかかる発明によれば、環境に応じた数の格子点を選択するので、ユーザが煩わしい設定作業を行わなくても良くなる。
【００４０】さらに、請求項７にかかる発明によれば、格子点の増加を変換画像に応じて選択するようにしているため、不要に色変換テーブルを大きくしすぎたり、不十分にしか色変換テーブルを大きくしていないなどといった不具合を無くすることができる。
【００４１】さらに、請求項９にかかる発明によれば、上述したのと同様に、より融通性の高い色変換テーブル生成することが可能な色変換テーブルの製造方法を提供することができる。
【００４２】さらに、請求項１９にかかる発明によれば、上述したのと同様に、より融通性の高い色変換テーブル生成することが可能な色変換テーブルの製造プログラムを、コンピュータ等で読み取り可能に記録した記録媒体として提供することができる。
【００４３】
【発明の実施の形態】以下、図面にもとづいて本発明の実施形態を説明する。
【００４４】図１は、本発明の一実施形態にかかる画像処理システムをブロック図により示しており、図２は具体的ハードウェア構成例をブロック図により示している。
【００４５】同図において、画像入力装置１０はカラー画像を撮像するなどして階調表色データを画像処理装置２０へ出力し、同画像処理装置２０は所定の画像処理を行なって画像出力装置３０に出力し、同画像出力装置３０は元のカラー画像を表示する。
【００４６】ここにおいて、画像入力装置１０の具体例はスキャナ１１やデジタルスチルカメラ１２などが該当し、画像処理装置２０の具体例はコンピュータ２１とハードディスク２２などからなるコンピュータシステムが該当し、画像出力装置３０の具体例はプリンタ３１やディスプレイ３２等が該当する。また、本発明をコンピュータ等に実施させるプログラムを記録可能な記録媒体は、ドライブ装置２３によってコンピュータに読み込まれるＣＤ－ＲＯＭ２４等の記録媒体が相当する。
【００４７】画像入力装置１０としてのスキャナ１１が階調表色データとして例えばＲＧＢ（緑、青、赤）の階調データを出力するものとするとともに、画像出力装置３０としてのプリンタ３１は階調表色データとしてＣＭＹ（シアン、マゼンダ、イエロー）の二値データを入力として必要とするものとすると、画像処理装置２０としてのこのコンピュータ２１の具体的役割は、ＲＧＢの階調データをＣＭＹの二値データに変換することである。また、ディスプレイ３２がＲＧＢの階調データを入力するものとしても、スキャナ１１とディスプレイ３２では色特性が異なるのが通常であり、コンピュータ２１はＲＧＢの階調データをＲＧＢの階調データに変換する処理を行なうことになる。デジタルスチルカメラ１２についてもほぼ同様のことがいえる。
【００４８】このコンピュータ２１の内部で行なわれる処理を図３に示している。図に示すように、コンピュータ２１内ではオペレーティングシステム２１ａが稼働しており、プリンタ３１やディスプレイ３２に対応したプリンタドライバ２１ｂやビデオドライバ２１ｃが組み込まれている。一方、アプリケーション２１ｄはオペレーティングシステム２１ａにて処理の実行を制御され、必要に応じてプリンタドライバ２１ｂやビデオドライバ２１ｃと連携して所定の処理を実行する。
【００４９】アプリケーション２１ｄで生成される印刷用データはオペレーティングシステム２１ａを介してプリンタドライバ２１ｂに入力され、当該プリンタドライバ２１ｂはプリンタ３１が要求するフォーマットの画像

データに変換する。この変換が上述したＲＧＢの階調データをＣＭＹの二値データに変換する処理に該当する。ここにおいて、同プリンタドライバ２１ｂは、アプリケーション２１ｄが所定の画面単位で生成する画像データからプリンタ３１における印刷ヘッドの走査範囲を切り出すラスタライザ２１ｂ１と、この走査範囲の各画素について色変換テーブルを参照してＲＧＢの階調データをＣＭＹの階調データに変換する色変換部２１ｂ２と、ＣＭＹの階調データを二値データに階調変換する階調変換部２１ｂ３とから構成されている。なお、アプリケーション２１ｄが生成する表示画像データについてはビデオドライバ２１ｃが所定の画面用メモリに書き込み、ハードウェア回路を介してディスプレイ３２にて表示させている。

【００５０】色変換部２１ｂ２は色補正モジュールとも呼ばれ、色変換の演算処理を実行する色変換用ソフトウェア２１ｂ２ａと色変換テーブル２１ｂ２ｂとから構成されている。色変換テーブル２１ｂ２ｂは異なる表色空間の間で階調表色データを変換するために変換元の表色空間での格子点に変換先の表色空間での階調表色データを対応させたものであり、より具体的には図４に示すような三次元のＲＧＢ階調データを座標値としてＣＭＹ階調データを読み出すための三次元ルックアップテーブルである。そして、色変換用ソフトウェア２１ｂ２ａは各画素のＲＧＢ階調データを座標値としてＣＭＹ階調データを読み出す処理を行う。

【００５１】この色変換部２１ｂ２を含めて同プリンタドライバ２１ｂは、図５に示すインストールプログラムによってハードディスク２２上に展開される。このインストーラは、機器チェックを行なうステップＳ１１０と、上記色変換用ソフトウェア２１ｂ２ａを含むドライバ用ソフトウェアをハードディスク２２上に展開するステップＳ１２０と、所定の補間演算によって小サイズの＊元色変換テーブル２１ｂ２ｃから所定のサイズの色変換テーブル２１ｂ２ｂを生成するステップＳ１３０とから構成されている。

【００５２】すなわち、補間演算によって小サイズの元色変換テーブル２１ｂ２ｃから所定のサイズの色変換テーブル２１ｂ２ｂを生成するステップＳ１３０こそが本発明における色変換テーブルの製造装置を構成し、その手順が色変換テーブルの製造方法を構成している。この具体的手法については後述するとして、本実施形態においては、プリンタドライバ２１ｂのインストーラとして具現化されているものの、その機能として小サイズの元色変換テーブル２１ｂ２ｃから所定のサイズの色変換テーブル２１ｂ２ｂを生成するものであればよい。従って、色変換テーブルを単独で生成するソフトウェアであってもよいし、あるいはワイヤーロジックからなるハードウェアなどで構成することも可能である。また、後述するように色変換用ソフトウェア２１ｂ２ａが必要に応じて色変換テーブル２１ｂ２ｂを生成するよう構成することも可能である。

【００５３】次に、この補間演算処理について詳述する。

【００５４】まず、補間演算処理の一例として非線形補間演算を採用する場合について説明する。

【００５５】ｎ個の点（Ｘｉ、Ｙｉ）（ｉ＝０，１，…，ｎ－１）が与えられればＹｉ＝Ｐ（Ｘｉ）（ｉ＝０，１，…，ｎ－１）を満たすｎ－１次の多項式

【００５６】

【数１】

$$P(X) = c_{n-1} \cdot X^{**(n-1)} + c_{n-2} \cdot X^{**(n-2)} + \cdots$$

【００５７】が一意的に定まる。但し、どの二つのＸｉも等しくないとする。この多項式を表す閉じた式

【００５８】

【数２】

$$P(X) = \sum_{i=0}^{N-1} \{ Y_i \prod_{j \neq i} ((X - X_j) / (X_i - X_j)) \}$$

【００５９】がラグランジュ（Ｌａｇｒａｎｇｅ）の補間公式である。なお、右辺のΠ以下については（（Ｘ－Ｘｊ）／（Ｘｉ－Ｘｊ））をｊ＝ｉ以外の全てのｊについて掛け合わせたものを意味する。この補間演算の具体的な実行方法をＣ言語で示したコーディングリストを図６に示している。

【００６０】さて、このような補間演算を用いれば、図７に示すような実際には各軸方向に五つの格子座標しか持たない小サイズの元色変換テーブル２１ｂ２ｃを使用して各格子座標間に新たな三つの格子座標を増加して格子座標間を四分することは可能である。この場合の格子座標間隔は「６４」階調に相当し、元色変換テーブル２１ｂ２ｃの各格子座標に格子番号「０」～「４」を付すものとする。また、新たに格子座標間を四分することに※より色変換テーブル２１ｂ２ｂでは格子番号「０」～「１６」となり、格子座標間隔は「１６」階調になる。なお、本来「２５６」階調としてこのように均等に分配する格子点を設けるのは不可能であり、計算上は「０」～「２５６」というような（実際の階調数＋１）の格子番号を想定しておき、計算の最後で最後の格子番号（例えば２５６）を現実の階調の範囲の最後の格子番号（２５５に相当）に移行させることによって計算を簡易にする。

【００６１】元色変換テーブル２１ｂ２ｃのインストールプログラム内での記憶フォーマットについて概略を触れておくと、図８の上段に示すように、ＲＧＢの各成分を座標値としてそれぞれＣＭＹの三色のデータを対応すべく、要素数が（５，５，５，３）の配列となってお

り、ファイルの先頭からベタで書き込まれている。従って、元色変換テーブル２１ｂ２ｃの対応データを参照するためにはＲ軸とＧ軸とＢ軸とのそれぞれに対応した格子番号ポインタＰｒ，Ｐｇ，Ｐｂを設定し、ファイルの先頭から（Ｐｒ×５×５×３＋Ｐｇ×５×３＋Ｐｂ×３）をオフセットアドレスとしてシアン（Ｃ）は「１」バイト目、マゼンダ（Ｍ）は「２」バイト目、イエロー（Ｙ）は「３」バイト目を読み出すことになる。

【００６２】むろん、かかる記憶フォーマットは一例に過ぎず、例えば、シアンについて全座標分だけ並べ、マゼンダ、イエローについて、順次、全座標分だけ並べていくなどの配置でも構わない。あるいは、ファイル圧縮した状態で保存してあっても良い。ただし、配列としてベタで書き込まれている場合には、後述するようにポインタ値で読み出しアドレスを演算でき、読み出す際の規則性を自由に設定できる。

【００６３】一方、色変換テーブル２１ｂ２ｂの記憶フォーマットを図８の下段に示している、上述したように元色変換テーブル２１ｂ２ｃの各格子点間を四分して新たに三つの格子座標を形成するものであるため、要素数が（１７，１７，１７，３）の配列となっており、ファイルの先頭からベタで書き込むようにしている。従って、この色変換テーブル２１ｂ２ｂの対応データを参照するためには先程と同様の格子番号ポインタＰｒ，Ｐｇ，Ｐｂを設定すると、ファイルの先頭から（Ｐｒ×１７×１７×３＋Ｐｇ×１７×３＋Ｐｂ×３）をオフセットアドレスとしてシアン（Ｃ）は「１」バイト目、マゼンダ（Ｍ）は「２」バイト目、イエロー（Ｙ）は「３」バイト目を読み出すことになる。

【００６４】色変換テーブル２１ｂ２ｂの格子点を増加する処理を図９のフローチャートに示している。この処理では、図８に示すように元色変換テーブル２１ｂ２ｃと色変換テーブル２１ｂ２ｂとで一致する格子点が存在するので、ステップＳ２１０にて格子点データを移転させる作業を行いつつ、格子点の間に空白のデータを挿入してハードディスク２２上にファイルの形として展開する。この後、ステップＳ２２０にて色変換テーブル２１ｂ２ｂの全格子点について対応データを埋めるべく、各軸の格子番号についてネストしたループ処理を実行する。格子番号は「０」～「１６」であるので、Ｒ軸、Ｇ軸、Ｂ軸についてポインターに「０」～「１６」を設定して処理を繰り返す。最も内側のループ内ではＲ軸、Ｇ軸、Ｂ軸のポインターで示される格子点が元色変換テーブル２１ｂ２ｃから移行した格子点に一致するか否かを判断し、一致しなければ補間演算で格子点の対応データを算出する処理を実行する。しかし、一致するのであれば既に対応データがあるので補間演算の処理をスキップする。

【００６５】補間演算の処理の一例を、図１０～図１２に示している。まず、フローを説明する前に図１１にて図示した非線形演算の概念を説明する。

【００６６】数２に示したラグランジュの補間公式を四点の対応データに基づいて適用しようとした場合、図１１に示すＰ点（Ｒｐ，Ｇｐ，Ｂｐ）の補間演算を行なうこととしても、必ずしも四つの格子点を通過するかどうかは不明である。従って、Ｐ点が位置する前後で各軸方向に四つの格子点からなる立方体を想定し、この立方体内で各軸方向ごとに順に補間演算を実行することにより、Ｐ点の演算に必要な四つの点の対応データを算出していくことにする。ここにおいて、各軸毎の格子座標を｛R1，R2，R3，R4｝｛G1，G2，G3，G4｝｛B1，B2，B3，B4｝と設定しておく。

【００６７】まず、Ｐ点（図示△の点）を通過するＧ軸方向に平行な直線を想定すると、この直線は、Ｇ軸の格子座標を通過することになる四つのＲＢ平面を貫通することになる。この各交点は同図にて○点で示しており、その座標は（Ｒｐ，Ｇ４，Ｂｐ）、（Ｒｐ，Ｇ３，Ｂｐ）、（Ｒｐ，Ｇ２，Ｂｐ）、（Ｒｐ，Ｇ１，Ｂｐ）である。この交点自体の対応データは不明であるため、それぞれの交点と交わるＲＢ平面上でＢ軸に平行な直線を想定する。この直線はＢ軸の格子座標を通過することになる四つのＲＧ平面を貫通する。四つの直線のうちＧ軸の座標が「Ｇ１」である点に注目し、各交点を同図にて●点で示している。その座標は（Ｒｐ，Ｇ１，Ｂ１）、（Ｒｐ，Ｇ１，Ｂ２）、（Ｒｐ，Ｇ１，Ｂ３）、（Ｒｐ，Ｇ１，Ｂ４）であり、まだ対応データは不明である。しかしながら、これらの交点を通過するＲ軸に平行な直線を想定すると、今度は全て格子点を通過する。すなわち、交点（Ｒｐ，Ｇ１，Ｂ１）を通過する直線は（Ｒ１，Ｇ１，Ｂ１）、（Ｒ２，Ｇ１，Ｂ１）、（Ｒ３，Ｇ１，Ｂ１）、（Ｒ４，Ｇ１，Ｂ１）を通過する。

【００６８】これを逆に遡ることにすれば、四つの（Ｒ１，Ｇ１，Ｂ１）、（Ｒ２，Ｇ１，Ｂ１）、（Ｒ３，Ｇ１，Ｂ１）、（Ｒ４，Ｇ１，Ｂ１）から一つの●点の対応データを得ることができることになり、同様にして四つの●点の対応データを得たときには一つの○点の対応データを得ることができる。これを繰り返せば四つの○点の対応データを得ることができ、そうなれば△点の対応データを算出できるようになる。

【００６９】かかる過程のより具体的な演算を図１２に示しており、一番内側のネストではｉ＝１～４とした四つの格子点の対応データＤ（Ｒｉ，Ｇｊ，Ｂｋ）を利用し、Ｒ軸方向での成分値Ｒｐでの対応データｆ（ｊ）（●点の対応データ）を算出している。ｊ＝１～４として四つのｆ（ｊ）が得られれば、一つ上のネスト内ではこれを利用してＢ軸方向での成分値Ｂｐでの対応データｇ（ｋ）（○点の対応データ）を算出する。そして、ｋ＝１～４として四つのｇ（ｋ）が得られれば、一番上のネスト内ではこれを利用してｈ（△点の対応データ）が算出できる。

【００７０】図１０に示すフローチャートに戻ると、ステップＳ３１０では所属格子グループの特定を実行する。図１１及び図１２に示すように各軸方向に四つの格子点を固定して演算を実行すると容易であるため、演算のルーチンをこの立方体の座標値を利用して実行できるサブルーチン化している。従って、格子点を補間する演算を実行する前に当該格子点を含むような各軸方向に四つの格子点からなる立方体を特定する。そして、ステップＳ３２０ではこの立方体の格子点における対応データを同ワークエリアへ移動させる。

【００７１】ワークエリアでは図１１に示す関係が特定されるため、続くステップＳ３３０では図１２に示すネスト処理で非線形演算を実行する。なお、ワークエリアへ移動させる際には各軸方向へのオフセットが生じるため、移動させる際にオフセット量を保存し、増加する格子点についてもその座標値に同オフセット量を考慮した座標値（Ｒｐ，Ｇｐ，Ｂｐ）で計算する。なお、図１２においては三次元での補間に対応して三段階のネストの処理となっているが、さらに高次元での補間に対応してネスト処理することも可能である。

【００７２】このようにして元色変換テーブル２１ｂ２ｃの格子点以外の格子点で非線形演算を施していくことにより、各軸のループ処理を終了したときに完全な色変換テーブル２１ｂ２ｂを得ることができる。

【００７３】上述した実施形態においては、非線形演算の具体的な処理としてラグランジュの補間公式を利用しているが、他の演算を利用することも可能であり、例えば、スプライン（ｓｐｌｉｎｅ）補間も可能である。スプライン補間は導関数まで連続性を有する利用ができ、この意味で導関数の連続性が問題となる場合に備えた硬めの補間である。ただし、計算は複雑とならざるを得ず、このスプライン補間演算の具体的な実行方法をＣ言語で示したコーディングリストを図１３に示している。

【００７４】また、他の非線形の補間演算として、ネビル（Ｎｅｖｉｌｌｅ）補間であったり、ニュートン（Ｎｅｗｔｏｎ）補間などを利用可能である。これらの場合は、数値的にも計算が楽になる。

【００７５】一方、上述した格子点については格子間隔を一定としている。これにより、特定の立方体をワークエリアとして対応データを移動させてしまい、演算を分かりやすく実行することができるようになる。しかしながら、必ずしも格子間隔が一定でなければならないわけではなく、格子間隔などを加味した係数などを利用して実行することも可能である。

【００７６】さらに、上述した実施形態においては、ステップＳ２２０の補間演算処理で図１０～図１２に示す非線形補間を利用しているが、図１４～図１９に示すように線形補間を利用することも可能である。

【００７７】図１４には格子点を増加する前の格子点位置を白丸で示すとともに、格子点を増加した後の格子点位置を黒丸で示しており、演算の簡易のため、格子間隔を半分にする位置に新たな格子点を設けている。従って、図に示す当初の格子番号は括弧書きのような丁度二倍の格子番号となる。なお、当初の格子点の数を仮に「ｉ」として説明する。また、図１５は線形補間の処理を実行するＣＰＵの手順をフローチャートにより示しており、図１６は最初の対応データの移動の状況を示しており、図１７は補間される格子点を示しており、図１８は補間演算の状況を示している。

【００７８】格子点間を半分とする格子点を各軸に形成するものとすると、補間前の格子点の格子座標は図１４の括弧書きに示すように自動的に（０，２，４，６，８…）となり、その間を補間することになる。図１５に示すフローチャートに戻ると、まず、ＣＰＵはステップＳ４１０にて既にテーブル内にある格子点データを新たなテーブルの所定位置に移行する処理を行う。例えば、図１６に示すように、格子座標（０，０，０）の対応データは新たなテーブルの格子座標（０，０，０）の対応データとして、格子座標（０，０，１）の対応データは新たなテーブルの格子座標（０，０，２）の対応データとして、格子座標（０，０，２）の対応データは新たなテーブルの格子座標（０，０，４）の対応データとしてというようにして移行していく。

【００７９】線形補間で格子点を補間する場合、周囲の八つの格子点からなる格子立法体内の位置によって演算が異なる。すなわち、辺上に存在する格子点の場合は両側の二点の格子点から補間されるし、面上に存在する格子点の場合は周辺の四つの格子点から補間されるし、中心に存在するものの場合は八つの格子点から補間される。

【００８０】格子点を増加する順序として、まず、ステップＳ４２０では格子辺上で格子点を生成する処理を実行する。ＣＰＵの演算処理では各軸毎にパラメータを与えてネストしたループで処理を行うため、図中においてもブロックを入れ子状に表示している。

【００８１】パラメータは各軸ともに「０」、「２」、「４」、「６」、「８」…と与え、Ｒ軸方向についていえば格子座標（１，０，０）の対応データを格子座標（０，０，０），（２，０，０）のデータから生成する。即ち、図１８に示すように、格子座標（０，０，０）の対応データＸ１と格子座標（２，０，０）の対応データＸ２とを足し、その結果Ｘ３を「２」で割ったものＸ４となる。ここにおいて「２」の除算は二進数データにおいて１ビットの右シフトに対応し、極めて容易に実行できる。むろん、最初に１ビットの右シフトを実行しておいてから足しても良く、この場合は演算過程でのオーバーフローを防止できる。以下、このパラメータの全組合せから格子辺上の格子点を生成する。

【００８２】ステップＳ４３０では格子面上で格子点を生成する処理を実行する。この場合もネストしたループ

で処理を行うため、各軸のパラメータとして「０」、「２」、「４」、「６」、「８」と与え、ＲＧ面と平行な面についていえば格子座標（１，１，０）の対応データを格子座標（０，０，０），（０，２，０），（２，０，０），（２，２，０）のデータから生成する。この場合は四つの格子点の平均値を取ることになり、四つのデータを足してから「４」で割ればよい。なお、「４」の除算は二進数データにおいて２ビットの右シフトに対応し、極めて容易に実行でき、以下、このパラメータの全組合せから格子面上の格子点は生成される。

【００８３】最後に、ステップＳ４４０では中心点の格子点を生成する処理を実行する。この場合は、各軸のパラメータとして「１」、「３」、「５」、「７」…と与え、格子座標（１，１，１）の対応データは周縁の八つの格子座標（０，０，０），（０，０，２），（０，２，０），（０，２，２），（２，０，０），（２，０，２），（２，２，０），（２，２，２）の対応データから生成する。この場合は八つの格子点の平均値を取ることになり、オーバーフローしないように３ビットの右シフトを実行してから足し合わせればよい。以下、このパラメータの全組合せから全中心点の格子点が生成される。

【００８４】以上の処理を行うことによって格子点の補間が終了する。本実施形態においては格子間隔を半分にするように格子点を増加させているが、この例に限らず、必要に応じて適宜増減可能であり、記憶資源の許容範囲内で格子点を増加させればよい。

【００８５】また、上述した線形補間演算でも格子点の格子間隔が一定となっているが、線形補間においては格子間隔を適宜変えたとしても演算の基本は両側の二点の対応データにしか過ぎないため、格子間隔を変えたとしても演算は容易である。従って、対応データの変化カーブが大きい部分では格子間隔を細かくすることにより、演算容易のまま補間精度を向上させることができる。

【００８６】一方、非線形補間においても線形補間においても、増加させる格子点を必ずしも一定とする必要はなく、必要に応じて変化させることも可能である。

【００８７】格子点を増加させる処理の一例として、図１９のフローチャートには、システムの機器構成に応じてテーブル生成するシステム対応格子点増加処理を示している。

【００８８】この例では、ステップＳ５１０にて演算能力を表すＣＰＵの種類の入力を行ない、ステップＳ５２０にて同様に演算速度を表すクロックの入力を行い、ステップＳ５３０にて演算能力や演算速度に影響するメモリ容量の入力を行い、ステップＳ５４０にて生成先のハードディスクの残り容量を入力する。

【００８９】そして、これらの組合せに対応して予め設定されているシステム対応テーブルをステップＳ５５０にて参照し、最も適切な格子点の数を読み出す。格子点の数を得られたらステップＳ５６０にて格子点増加処理を行う。このシステム対応テーブルに記憶されている格子点の数は、一般的な傾向として、演算能力や演算速度が速ければ格子の間隔が大きくなり、ハードディスクの残り容量が多ければ格子の間隔は小さくなるといったように設定しておけばよい。むろん、機器構成の入力要素はこれらに限るものではないし、その軽重も一定ではない。例えば、ハードディスクの残り容量が多い場合、キャッシュとの兼ね合いもあるもののフルサイズのテーブルを作成することも不可能ではない。

【００９０】この場合、ユーザの使用環境として印刷する対象が写真などのビットマップ系のデータが多いかあるいはドローデータ系のデータが多いかをステップＳ５４０とステップＳ５５０との間で問い合わせるようにしても良い。そして、ビットマップ系のデータが多いならば、写真などの色再現性に重きを置かれている環境を想定して格子点の数を大きくすれば良いし、ドローデータ系のデータが多いならばビジネスグラフなどの色再現性があまり重要でない環境を想定して格子点の数を小さくすれば良い。

【００９１】これまではインストール時に格子点の増加処理を行って色変換テーブル２１ｂ２ｂを生成するようにしているが、印刷時に必要なサイズの色変換テーブル２１ｂ２ｂをコンピュータ２１のＲＡＭ等の高速アクセス可能な記憶部に生成するようにしてもよい。通常時は小さな元色変換テーブル２１ｂ２ｃをハードディスクに格納しておき、印刷実行時に必要なサイズの色変換テーブル２１ｂ２ｂを高速アクセス可能なＲＡＭ上に作るメリットは非常に大きい。むろんこの場案もハードディスクの場合と同様に、ＲＡＭでの利用可能な残り容量や、出力する画像データの量、望まれる出力品質等を参酌しながら格子の間隔を設定して展開するサイズを決めればよい。

【００９２】図３に示すように、アプリケーション２１ｄが印刷する場合にはオペレーティングシステム２１ａを介してプリンタドライバ２１ｂが起動されるが、このときにファイルタイプがプリンタドライバ２１ｂに渡される。プリンタドライバ２１ｂではこの時のファイルタイプ（例えば、ｂｍｐなど）からビットマップ系であるのかドローデータ系であるのかを判断し、それに対応した格子点の数を設定して色変換テーブル２１ｂ２ｂを生成する。この格子点の数の大きさについては上述したインストーラによる場合と同様の傾向で設定すればよい。むろん、入力データの種類を判別する方法としてはこのようなファイルタイプだけに限らず、実際の入力データの色数が多いか少ないかなどによって判断しても良いし、オペレーティングシステムがオブジェクトの種類を判別してプリンタドライバに通知するようにしても良い。

【００９３】このように、画像処理装置２０を構成する

コンピュータ２１にてインストーラが実行されると、ステップＳ１３０で元色変換テーブル２１ｂ２ｃから色変換テーブル２１ｂ２ｂを生成するが、このときの格子点増加処理では、ステップＳ４３０におけるラグランジュの補間公式を利用した非線形補間演算で格子点を増加させたり、線形補間で格子点を増加させるなどし、また、その際に固定した格子点の数であっても良いし環境や入力画像に応じた格子点の数としてもよく、小さなサイズの元色変換テーブル２１ｂ２ｃから適切なサイズの色変換テーブル２１ｂ２ｂを生成することができる。

【図面の簡単な説明】

【図１】本発明の一実施形態にかかる色変換テーブルの製造装置を適用した画像処理システムのブロック図である。

【図２】同画像処理システムの具体的ハードウェア構成例のブロック図である。

【図３】コンピュータのソフトウェア構成を示すブロック図である。

【図４】三次元ルックアップテーブルの概念を示す図である。

【図５】インストールプログラムのフローチャートである。

【図６】ラグランジュの補間演算をＣ言語でコーディングした図である。

【図７】元色変換テーブルの格子座標を示す図である。

【図８】元色変換テーブルと色変換テーブルのファイル構成を示す図である。

【図９】格子点増加処理プログラムのフローチャートである。

【図１０】非線形補間プログラムのフローチャートである。

【図１１】ラグランジュの補間公式で非線形補間する場合の手順を示す概念図である。

【図１２】ラグランジュ補間演算に対応したフローチャートである。

【図１３】スプライン補間演算をＣ言語でコーディングした図である。

【図１４】格子点を増加する前後の格子座標を示す説明図である。

【図１５】線形補間の格子点増加処理プログラムのフローチャートである。

【図１６】元色変換テーブルと色変換テーブルのファイル構成を示す図である。

【図１７】補間される格子点の位置を示す概略説明図である。

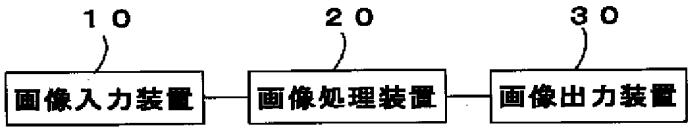
【図１８】ビットシフトを併用した演算の状態を示す説明図である。

【図１９】システム対応格子点増加処理プログラムのフローチャートである。

【符号の説明】

２０…画像処理装置
２１…コンピュータ
２１ａ…オペレーティングシステム
２１ｂ…プリンタドライバ
２１ｂ１…ラスタライザ
２１ｂ２…色変換部
２１ｂ２ａ…色変換用ソフトウェア
２１ｂ２ｂ…色変換テーブル
２１ｂ２ｃ…元色変換テーブル
２１ｂ３…階調変換部
２１ｃ…ビデオドライバ
２１ｄ…アプリケーション
２２…ハードディスク
２３…ドライブ装置
２４…ＣＤ－ＲＯＭ

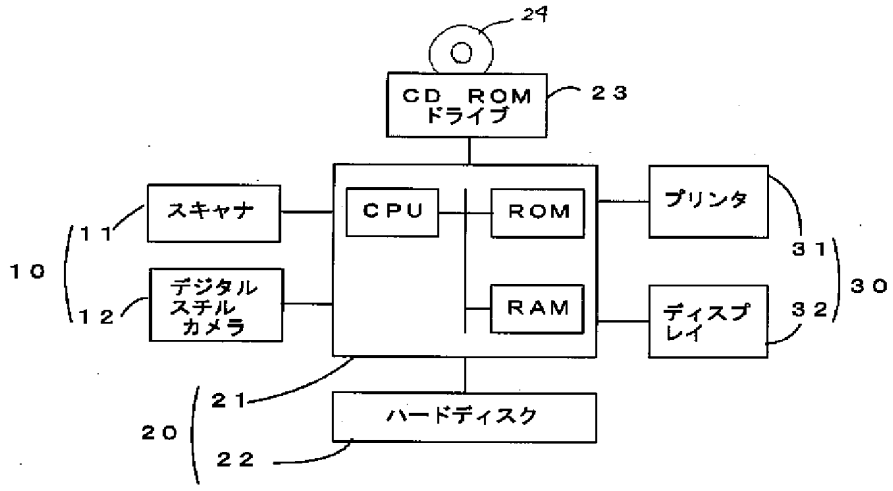
【図１】



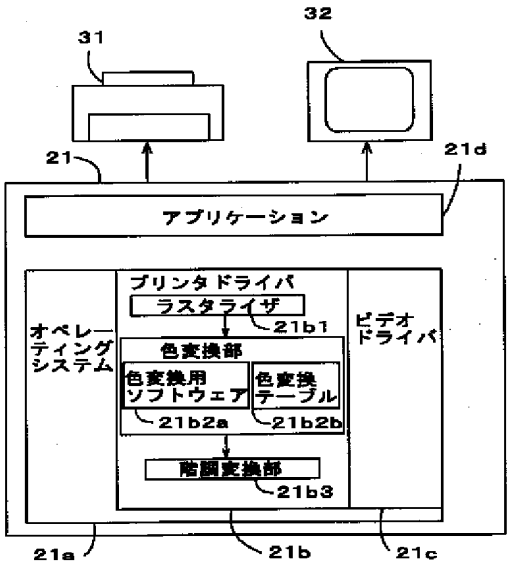
【図４】

Ｃｘ（Ｒｘ，Ｇｘ，Ｂｘ）
Ｍｘ（Ｒｘ，Ｇｘ，Ｂｘ）
Ｙｘ（Ｒｘ，Ｇｘ，Ｂｘ）

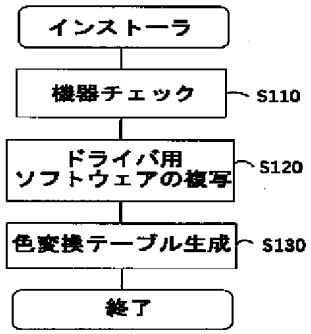
【図２】



【図３】



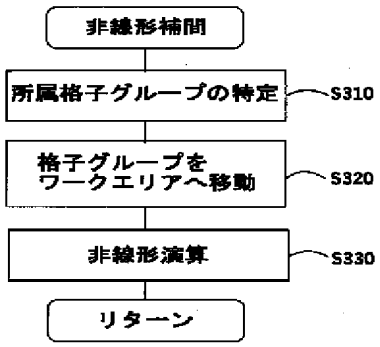
【図５】



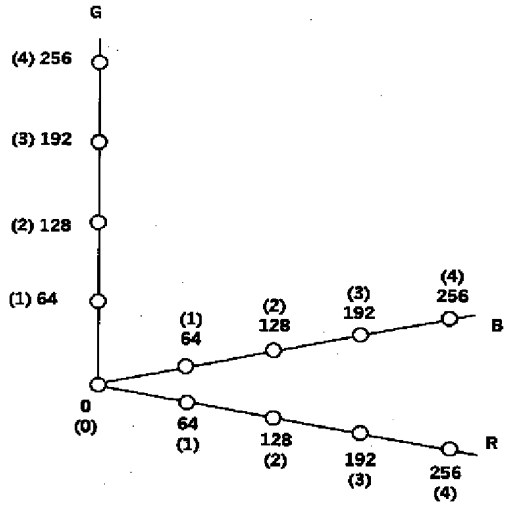
【図６】

```
1 double lagrange(double t)
2 {
3       int I,J;
4       double sum,prod;
5
6       sum=0;
7       for (I=0;I<N;I++) {
8             prod=y[I];
9             for (J=0;J<N;J++)
10                  if (J!=I) prod ＊ =(t-x[J])/(x[I]-x[J]);
11            sum +=prod;
12      }
13      return sum;
14 }
```

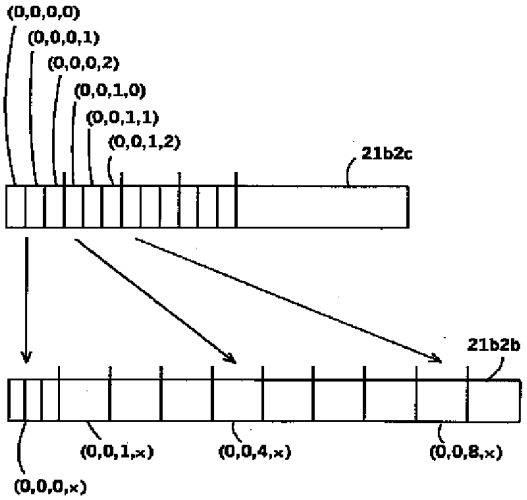
【図１０】



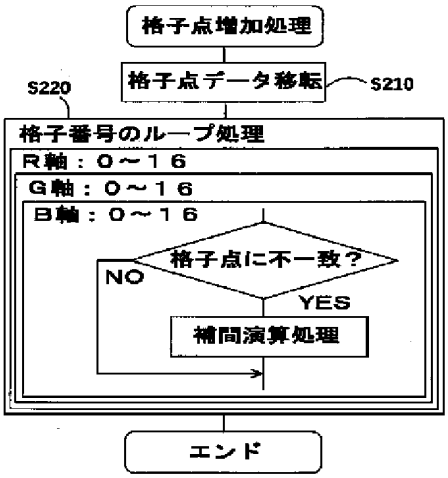
【図７】



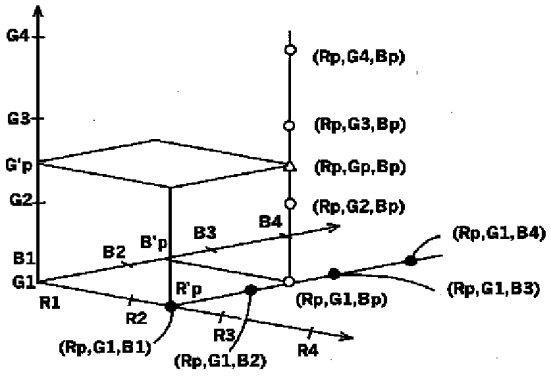
【図８】



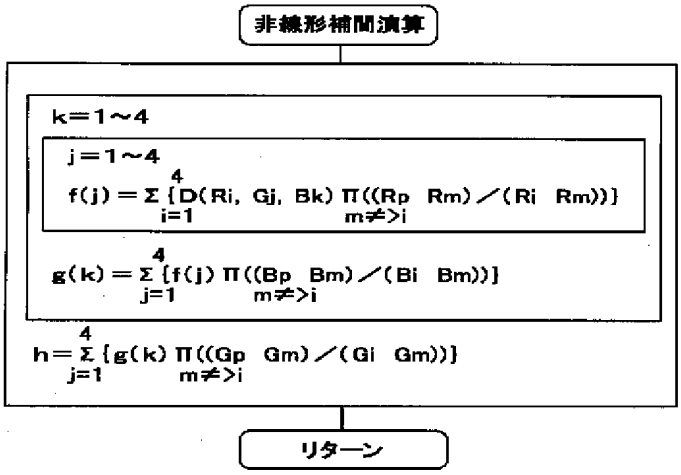
【図９】



【図１１】



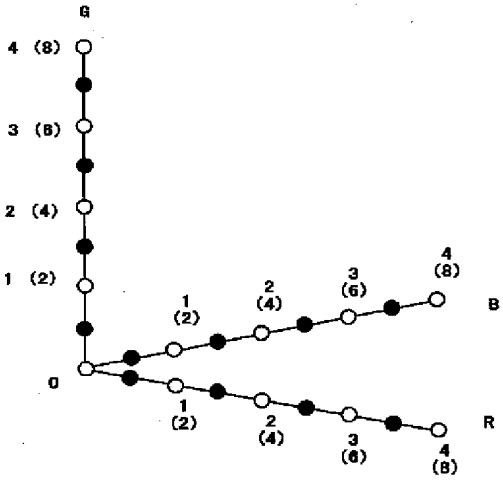
【図１２】



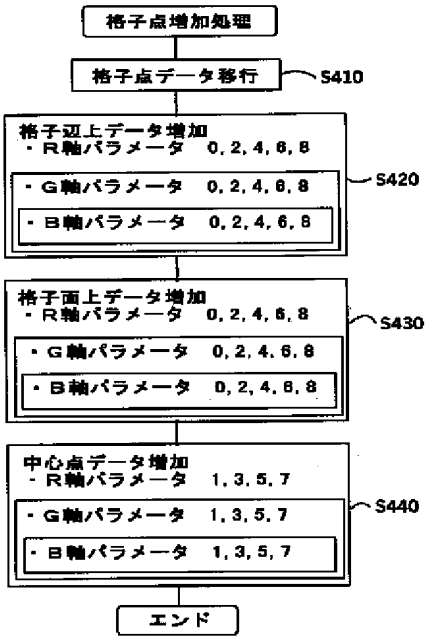
【図１３】

```
1 void maketable(double x[  ], double y[  ], double z[  ])
2 {
3       int i;
4       double t;
5       static double h[N], d[N];
6
7       z[0]=0; z[N-1]=0;          /＊  両端点でのｙ N(x)/6  ＊/
8       for (i=0; i < N-1; i++)  {
9               h[i    ]=x[i+1]-n[i];
10              d[i+1]=(y[i+1]-y[i])/h[i];
11      }
12      z[1]=d[2]-d[i]-h[0] ＊ z[0];
13      d[1]=2 ＊ (n[2]-x[0]);
14      for(i=1; i < N-2;i++)  {
15              t=h[i]/d[i];
16              z[i+1]=d[i+2]-d[i+1]-z[i] ＊ t;
17              d[i+1]=2 ＊ (x[i+2]-x[i])-h[i] ＊ t;
18      }
19      z[N-2]=h[N-2] ＊ z[N-1];
20      for(i=N-2;i > 0;i--)
21      z[i]=(z[i]-h[i] ＊ z[i+1])/d[i];
22 }
23
24 double spline(double t, double x[  ], double y[  ], double z[
25 {
26      int i, j, k;
27      double d, h;
28
29      i=0; j=N-1;
30      while (i < j)  {
31              k=(i+j)/2;
32              if(x[k] < t)  i=k+1;   else j=k;
33      }
34      if(i > 0) i--;
35      h=x[i+1]-x[i]; d=t-x[i];
36      return ((z[i+1]-z[i]) ＊ d/h+z[i] ＊ 3) ＊ d
37              +((y[i+1]-y[i])/h
38              -(z[i] ＊ 2 ＊ z[i+1] ＊ h)) ＊ d+y[i];
39 }
```

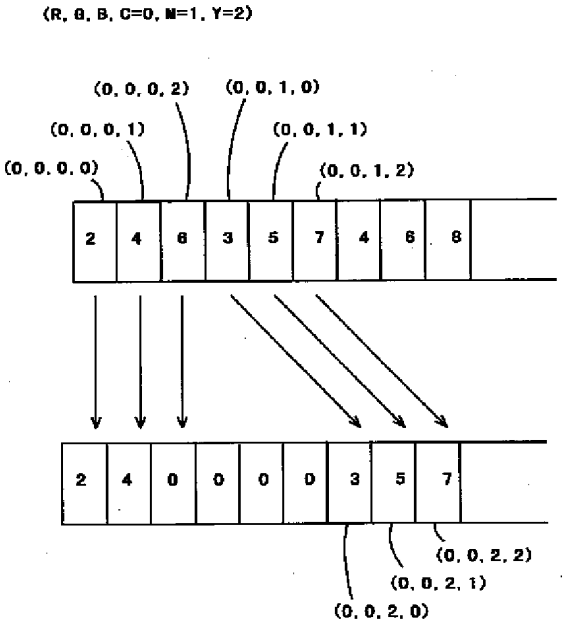
【図１４】



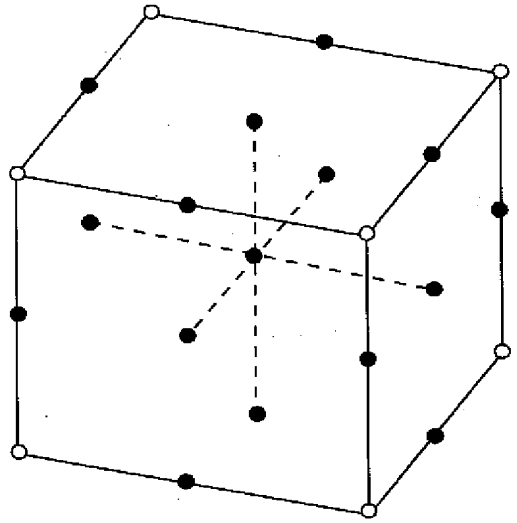
【図１５】



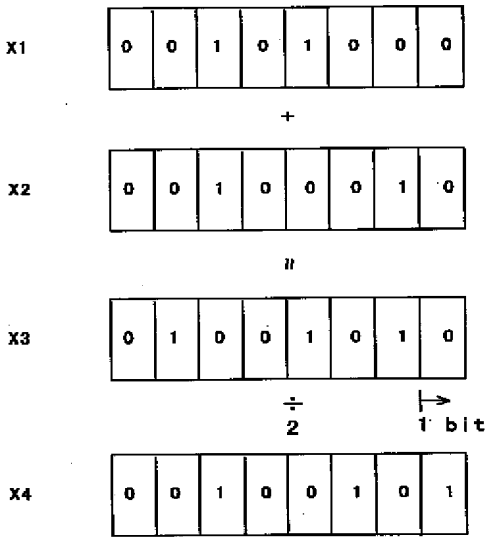
【図１６】



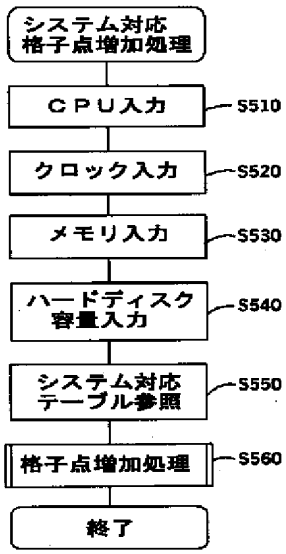
【図１７】



【図１８】



【図１９】



フロントページの続き

(51)Int.Cl.[6]	識別記号	ＦＩ
		Ｈ０４Ｎ　1/46　　Ｚ

PAT-NO: JP410191090A

DOCUMENT-IDENTIFIER: JP 10191090 A

TITLE: APPARATUS AND METHOD FOR MANUFACTURING COLOR CONVERSION TABLE, AND RECORDING MEDIUM

PUBN-DATE: July 21, 1998

INVENTOR-INFORMATION:

NAME	COUNTRY
FUKAZAWA, KENJI	
KASAHARA, HIROKAZU	

ASSIGNEE-INFORMATION:

NAME	COUNTRY
SEIKO EPSON CORP	N/A

APPL-NO: JP09274697

APPL-DATE: October 7, 1997

INT-CL (IPC): H04N001/60 , B41J002/525 , G06T001/00 , G09G005/06 , H04N001/46

ABSTRACT:

PROBLEM TO BE SOLVED: To generate a color conversion table optimum to the environment of the user or the like.

SOLUTION: When the installer is executed on a computer 2 being a component of an image processing unit, a color conversion table 21b2b is generated from an original color conversion table, number of grating points is increased by a nonlinear interpolation arithmetic operation or a linear interpolation arithmetic operation by utilizing the Lagrange's interpolation formula. In this case, number of the grating points may be fixed or number of grating points is selected in response to the environment or an input image and the color conversion table 21b2b with a proper size is generated from the original color conversion table with smaller size.